GISC2020
Global ICT Standards Conference

뉴노멀시대 선도를 위한
ICT 표준의 역할

OCF Core Framework 2.2.0

ETRI, 이주철
2020-10-16

GISC2020
Global ICT Standards Conference

뉴 노멀 시대
선도를 위한
ICT 표준의
역할

Contents

- ❑ OCF Core Framework
  - OCF Core Framework features
  - OCF Core Framework 2.2.0 issues
- ❑ IoTivity-Lite
  - Current Status

뉴 노멀 시대
선도를 위한
ICT 표준의
역할

OCF Core Framework

- OCF Core Framework features
- OCF Core Framework 2.2.0 issues

OCF Core Framework?

- Definition in spec.
  - The OCF **core architecture**, **interfaces**, **protocols** and **services** to enable OCF profiles implementation for Internet of Things (IoT) usages and ecosystems.

- Core Framework is divided into..
  - Mandatory: mandatory for all OCF Devices to implement
    - RESTful architecture, Resource model, Resource Discovery, Basic operations, Protocol stack, etc.
  - Optional: optionally implemented by any OCF Devices
    - Device/Platform configuration, Scene, Rule, etc.
  - Extension: optional functionalities that are significant in scope
    - e.g. Wi-Fi easy setup, Cloud

Design goal of OCF Core Framework

- Enables development of various vertical profiles (e.g. smarthome, health, etc.) on top of “core framework” while maintaining fundamental interoperability
- Scalable from resource constrained devices to resource rich devices
- Reuse open standard solutions (e.g. IETF) where they exist
- Alignment with IoTivity-Lite open source releases

Core Framework (Mandatory) : RESTful Architecture

- Apply RESTful architecture to IoT world
  - All Physical & Logical entities are described as “Resource”
    - Real world “things” are described as “Resource”
  - All Resources are accessed through “URI”
  - All Resources are manipulated through 5 operations (CRUD + N) with 2 Messages (Request/Response)

- 2 Roles in OCF world
  - OCF Client: A logical entity that accesses a Resource on a Server
    - Initiates a transaction (send a request)
  - OCF Server: A logical entity that exposes hosted Resources
    - Hosts resources
    - Sends response

Core Framework (Mandatory) : Resource Model

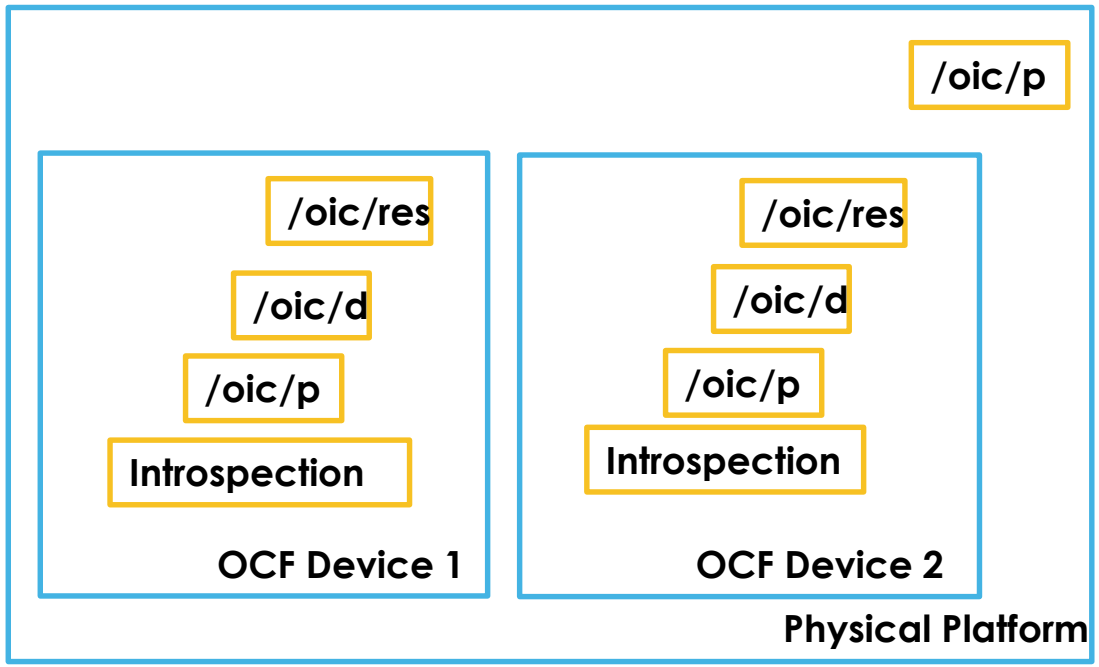
- How is OCF resource described?
  - **Properties (Identifier + other Properties)**
    - Identifier: “oic.r.temperature”
    - Properties: “temperature”, “units” …
  - Each Resource is described as “OAS (Open API Specification)”
- How can we access each Resource?
  - use URI: **ocf://<authority>/<path>?<query>**
  - e.g. ocf://<device id>/room/temperature?units=C
- Interface – a view into the Resource
  - OCF interface defines permissions on that view of the Resource

```
/room/temperature
{
    “rt” : [“oic.r.temperature”],
    “if” : [“oic.if.a”, “oic.if.s”, “oic.if.baseline”],
    “temperature”: 20.0,
    “units”: “C”
}
```

Core Framework (Mandatory) : Resource Model

❑ Concept of OCF Device & Platform

- Device: a logical entity that assumes one or more roles, e.g., Client, Server
- Platform: a Physical Device containing one or more Devices



Source: OCF_2.2.0_Specification_Overview.pptx

Core Framework (Mandatory) : Endpoint

- An (OCF) Endpoint is defined as the source or destination of a request and response messages for a given Transport Protocol Suites (e.g. CoAP over UDP over IPv6).
  - OCF Endpoint is identified by an IPv6 address and port number
- An OCF Device can be associated with multiple Endpoints
  - Each Resource can have different Endpoint
- An Endpoints can be shared among multiple OCF Devices
  - only when there is a way to indicate target Resource with Request URI

뉴 노멀 시대
선도를 위한
ICT 표준의
역할

Core Framework (Mandatory) : Endpoint

/oic/res


```
[
 { "href": "/oic/res",
   "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989/oic/res",
   "rel": "self",
   "rt": ["oic.wk.res"],
   "if": ["oic.if.ll", "oic.if.baseline"],
   "p": {"bm": 3},
   "eps": [{"ep": "coaps://[fe80::b1d6]:44444"}] },
 { "href": "/oic/p",
   "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
   "rt": ["oic.wk.p"],
   "if": ["oic.if.r", "oic.if.baseline"],
   "p": {"bm": 3},
   "eps": [{"ep": "coap://foo.bar.com:44444"}, {"ep": "coaps://foo.bar.com:11111"}] },
 { "href": "/oic/d",
   "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
   "rt": ["oic.wk.d", "oic.d.light"],
   "if": ["oic.if.r", "oic.if.baseline"],
   "p": {"bm": 3},
   "eps": [{"ep": "coap://[fe80::b1d6]:44444"}, {"ep": "coaps://[fe80::b1d6]:11111"}] },
 { "href": "/myLight",
   "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
   "rt": ["oic.r.switch.binary"],
   "if": ["oic.if.a", "oic.if.baseline"],
   "p": {"bm": 3},
   "eps": [{"ep": "coaps://[fe80::b1d6]:44444"}, {"ep": "coaps://foo.bar.com:11111"}] }
]
```


Endpoint for each target resource.

Source: OCF_2.2.0_Specification_Overview.pptx

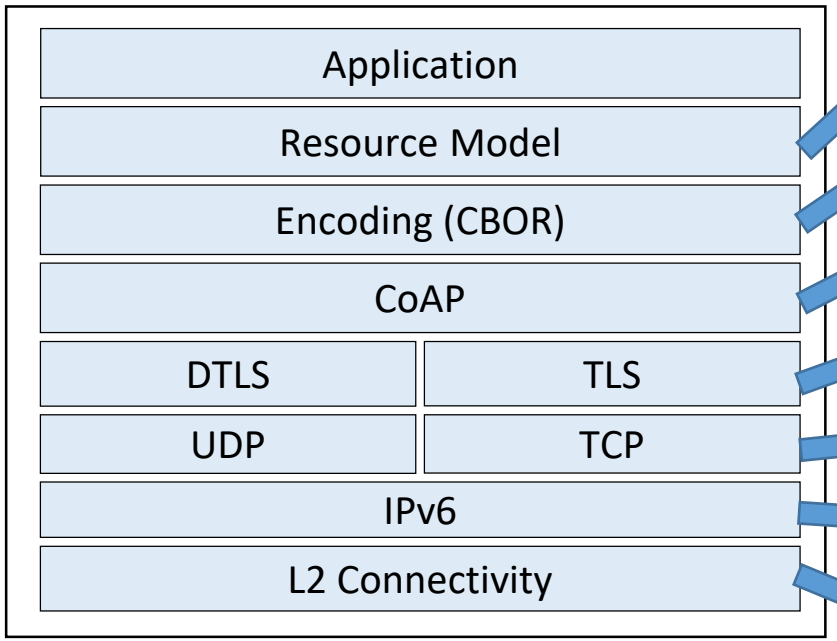
Core Framework (Mandatory) : Resource Discovery

- OCF devices make use of CoAP Discovery using IANA defined OCF Service Address (not the default CoAP address).
- Multicast RETRIEVE (CoAP GET) sent to well known URI (/oic/res)
- Response is an array of links; each link represents a Resource hosted by the responding server
  - Links provide:
    - href
    - Relationship (self link, hosted link, bridged link)
    - Endpoint binds
    - Supported interfaces
    - Observability of the Resource
- Different response views on /oic/res can be realized by using different supported interfaces with the RETRIEVE operation (e.g., CoAP GET /oic/res?if=oic.if.baseline, CoAP GET /oic/res?if=oic.if.b)

Core Framework (Mandatory) : Atomic Measurement Resource

- An OCF Resource that ensures a Client can only access the Properties of linked Resources (specified as OCF Links) atomically, as a whole, and read-only, using the “batch” interface
  - Atomically, meaning the value of all properties of the Atomic Measurement are sampled at the same time
  - As a whole, meaning that the values of all properties of the Atomic Measurement will be returned, or no value will be returned
  - Read-only, meaning that the properties of the Atomic Measurement can only be read, not written, using the batch interface. Any attempt to write to any property of the Atomic Measurement will result in an error.
- The primary example of Atomic Measurement Resources are with Healthcare vertical defined OCF Resources (e.g blood pressure measurement)

Core Framework (Mandatory) : Protocol Stack



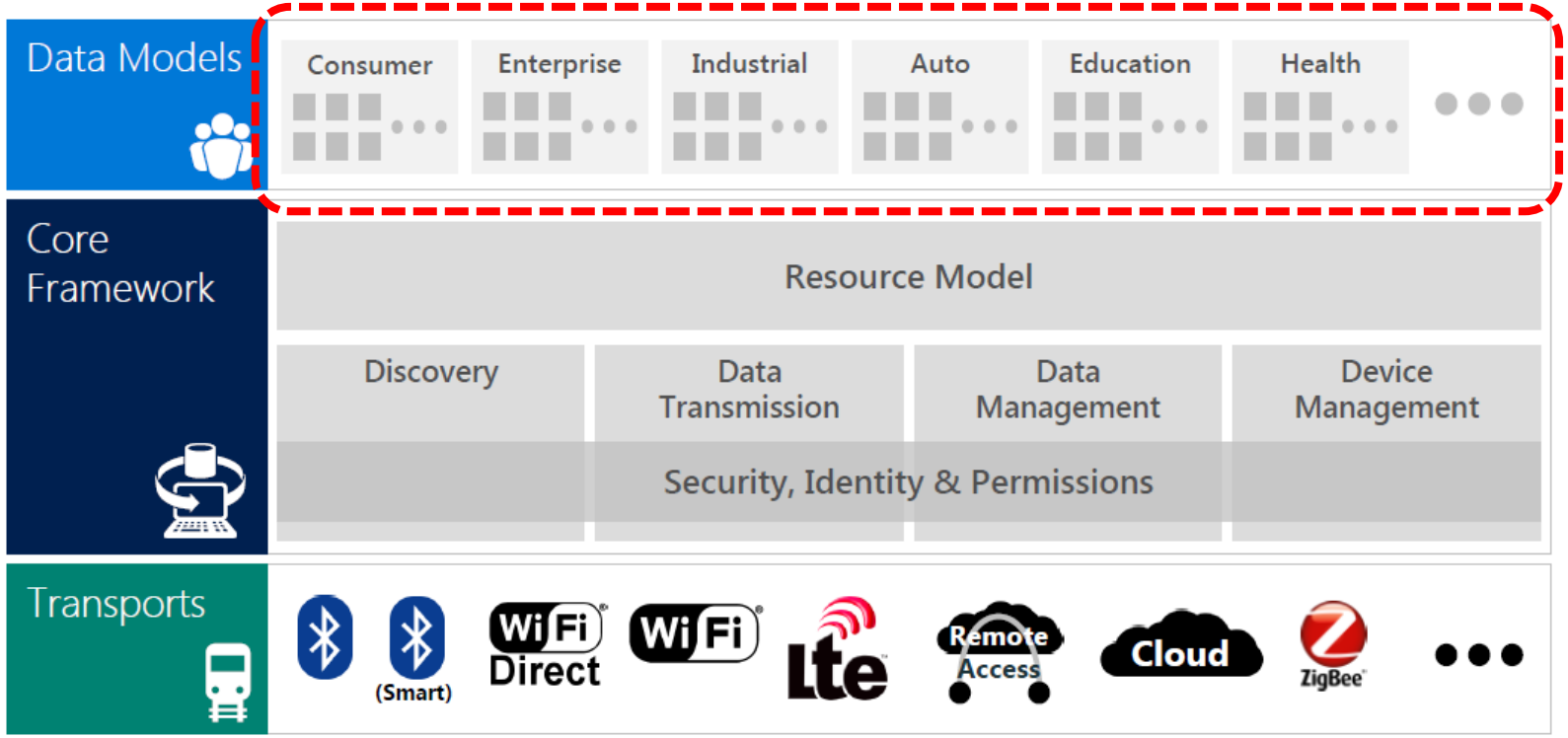
OCF Stack

- Common Resource Model defined using Open API 2.0
- Concise representation on the wire (binary encoded JSON)
- Constrained device support via use of CoAP as the transport layer
- Secure connection
- Connectionless or Connection Oriented
- IPv6 as the harmonization layer
- Agnostic of underlying physical layer technology (Wi-Fi, Ethernet, Thread)

Source: OCF_2.2.0_Specification_Overview.pptx

뉴 노멀 시대
선도를 위한
ICT 표준의
역할

Core Framework (Mandatory) : Protocol Stack



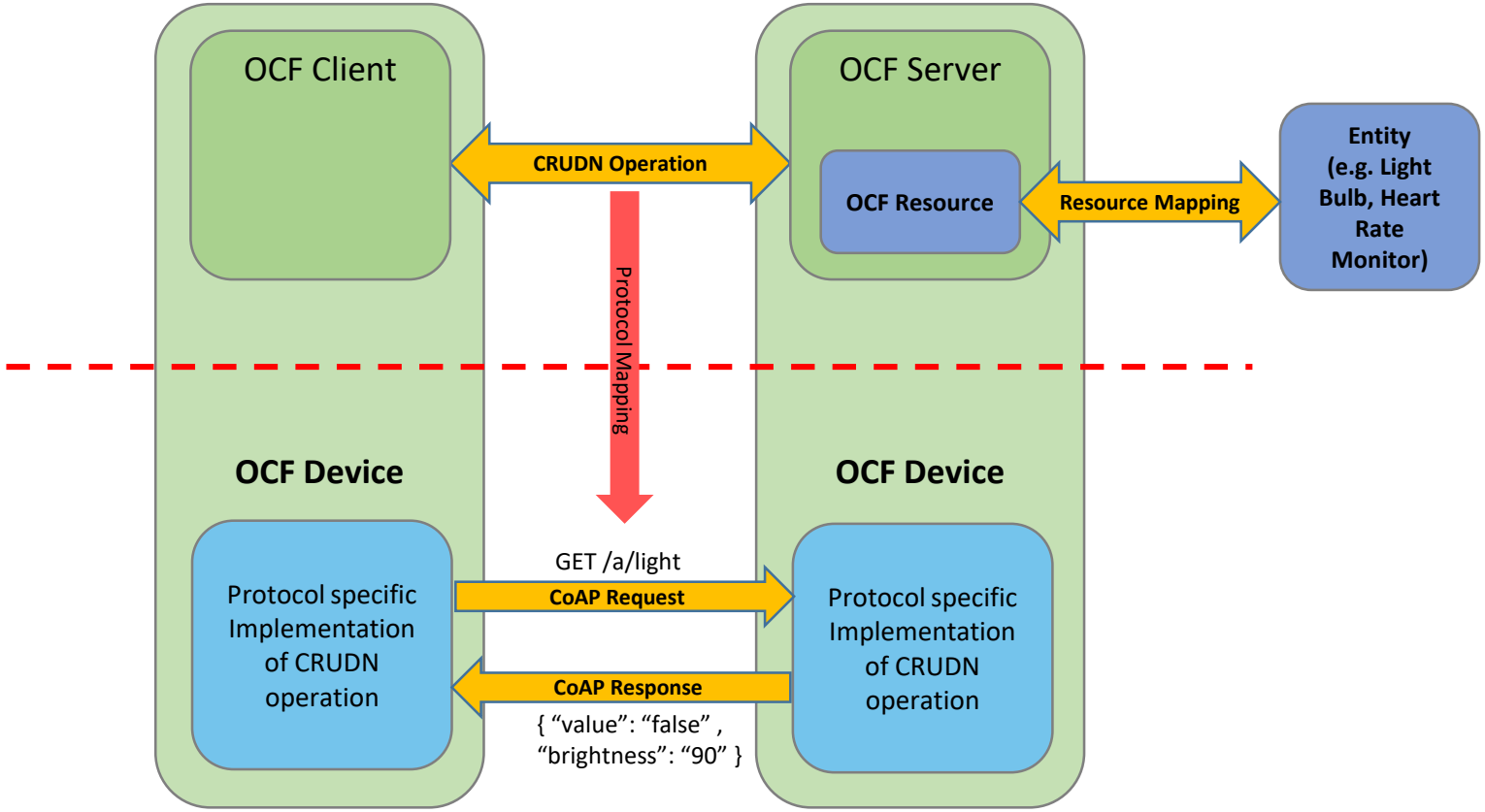
From *OCF Overview, Matt Perry*

Core Framework (Mandatory) : Typical Communication Flow

- Step 1. Resource Discovery
  - Send “Retrieve” multicast(or unicast) message for special resource (“**/oic/res**”)
  - Get
    - Server’s info (address, protocol, port, …)
      - ✓ e.g.) { fe80::187d:5945:e139:bb64, IPv6, 41243, … }
    - All resources that server has
      - ✓ e.g.) { … { “href”: “/a/temperature”, “rt”: “oic.r.temperature”,… }, … }
- Step 2. Manipulate Resources
  - Create: PUT /a/switch
  - Retrieve: GET /a/temperature?units=C
  - Update: POST /a/temperature { “temperature” : 29 }
  - Delete: DELETE /a/switch
  - Notify: GET /a/temperature { obs parameter = true }

뉴 노멀 시대
선도를 위한
ICT 표준의
역할

Core Framework (Mandatory) : All Together



Core Framework (Optional) : Device Management

- OCF Resources for maintaining & providing diagnostics info of a Device
  - Maintenance Resource (“oic.wk.mnt”)
    - Factory Reset, Reboot, Last Error
  - Network monitoring Resource (“oic.wk.nmon”)
    - Network indicator, reset, TX bytes, RX bytes, …
  - Software update Resource (“oic.r.softwareupdate”)
    - Resource to support secure software updates for Devices

Core Framework (Optional) : Scene & Rules (2.2.0)

- Scene
  - Scene is a mechanism for automating certain operations
  - Scenes provide a mechanism to store a setting over multiple Resources that may be hosted by multiple separate Servers
  - Scenes, once set up, can be used by multiple Clients to recall a setup
  - Related Resources
    - "oic.wk.scenelist" (collection)
      - ✓ "oic.wk.scenecollection" (collection)
        - "oic.wk.scenemember"
  - Example scene collection for "my living room"

```
/mySceneCollection4LivingRoom
{
    "sceneValues": [ "Off", "Reading", "TVWatching" ],
    "links": [
        { "href": "/scenemember1", … },
        { "href": "/scenemember2", … } ]
}
```

```
/scenemember1
{
    "link": {
        "rt":   ["oic.r.switch.binary"],
    },
    "SceneMappings": [
        {
            "scene":  "off",
            "memberProperty":  "value",
            "memberValue":     "false"
        },
        {
            "scene":  "Reading",
            "memberProperty":  "value",
            "memberValue":     "false"
        },
        …..
    ]
}
```

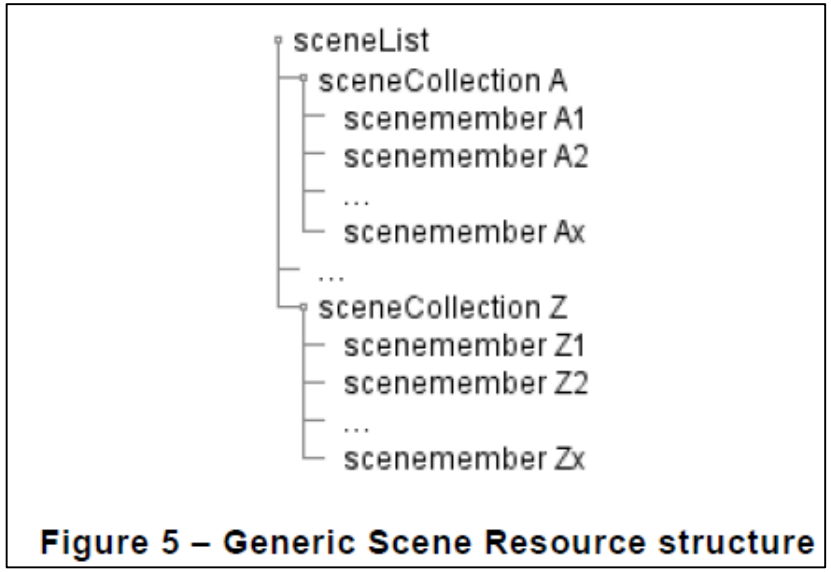
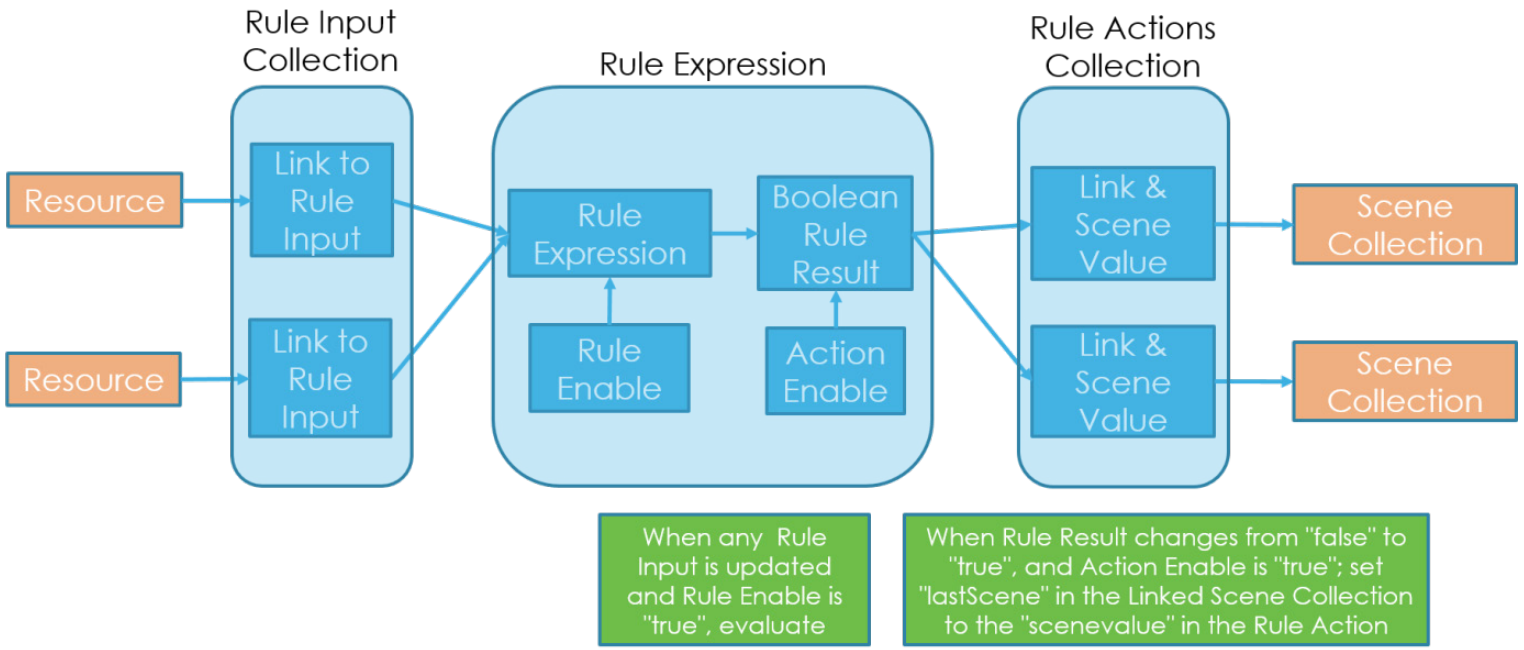


Figure 5 – Generic Scene Resource structure

Source: OCF Core Optional Specification v2.2.0

Core Framework (Optional) : Scene & Rules (2.2.0)

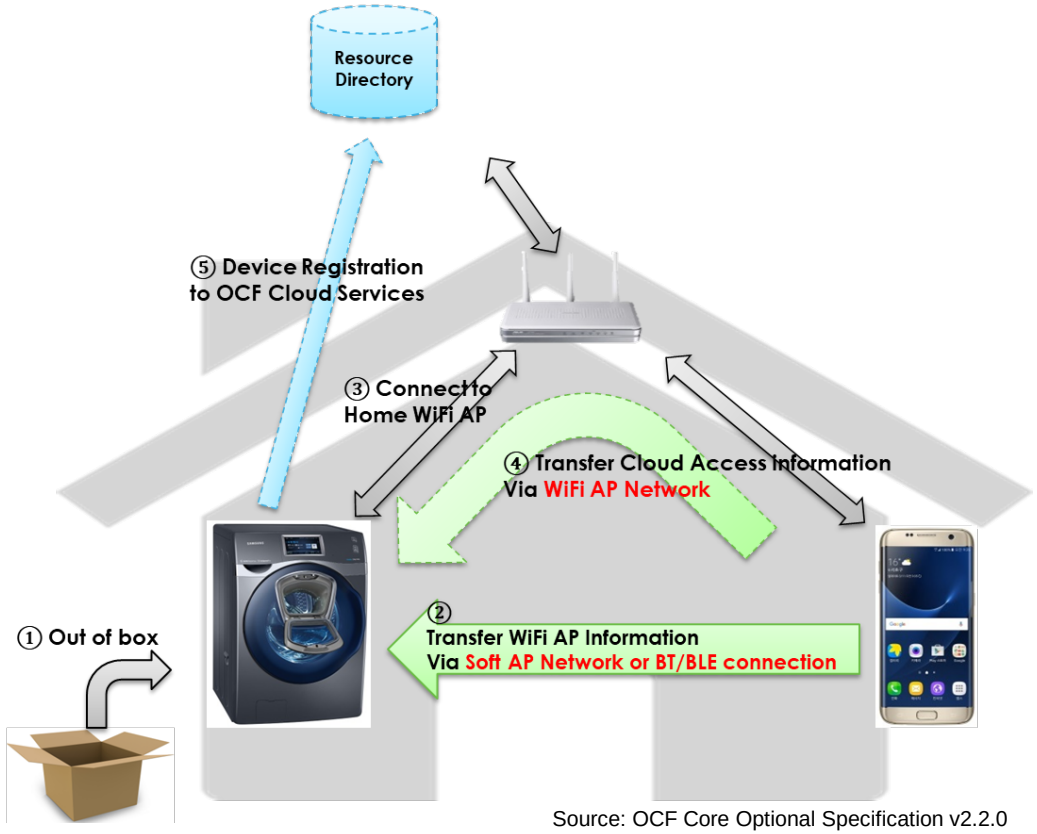
- Rules
  - Rules are Resources that implement autonomous decision logic according to a condition-action pattern
  - Related Resources
    - "oic.r.rule" (collection)
      - ✓ "oic.r.rule.inputcollection" (collection)
      - ✓ "oic.r.rule.expression"
      - ✓ "oic.r.rule.actioncollection" (collection)
        - "oic.r.rule.action"



Source: OCF Core Optional Specification v2.2.0

Core Framework (Extension) : WiFi Easy Setup

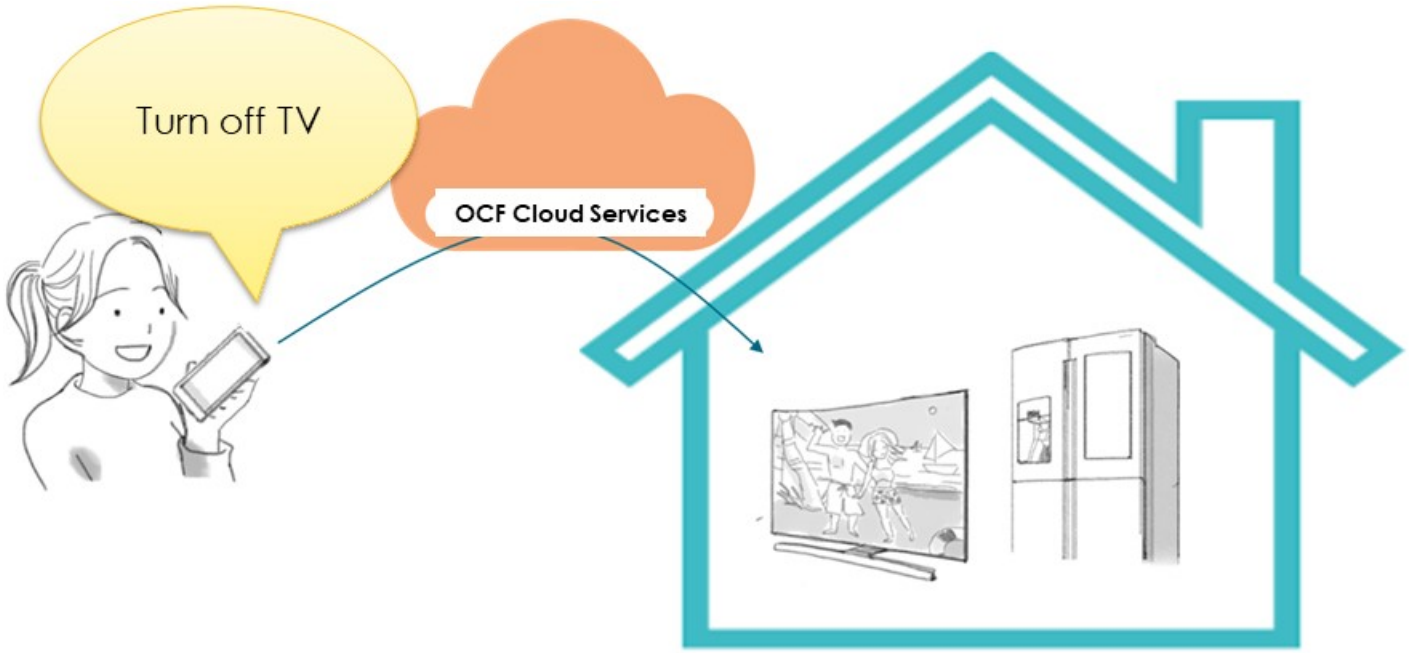
- WiFi Easy Setup is the 1st step when a device is unboxed. Specifically for UI-Less devices this is very important step. Wi-Fi Easy Setup spec defines interoperable data model that can be used to configure Wi-Fi connection on a device using a common communication channel. It also provides a standard way of a device proximally advertising its presence for discovery by clients that will perform the configuration. Other than Wi-Fi connection setup, OCF 2.0 specifications optionally provide a way to configure a connection with OCF Cloud Services.



Source: OCF Core Optional Specification v2.2.0

Core Framework (Extension) : Device to Cloud Services

- Remote Control/Manage OCF devices based on user authentication
  - User can access OCF devices which belong to or are shared with them regardless of location.
  - User can receive cloud services in conjunction with the registered devices

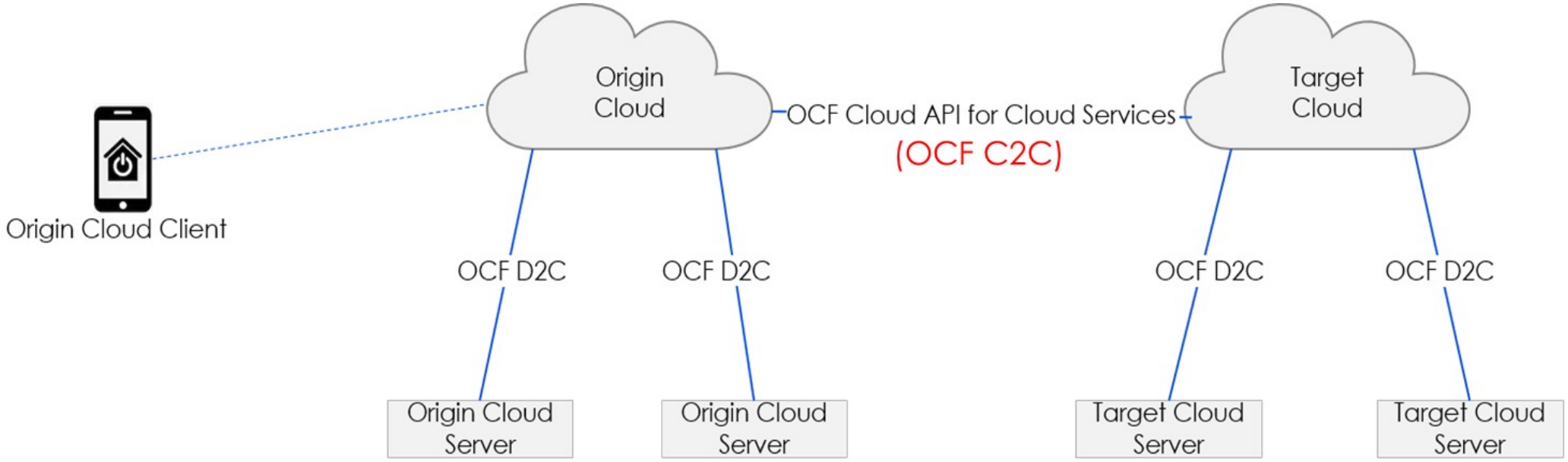


Source: OCF_2.2.0_Specification_Overview.pptx

뉴 노멀 시대
선도를 위한
ICT 표준의
역할

Core Framework (Extension) : Cloud API for Cloud Service (2.2.0)

- OCF C2C API enables user to connect OCF Devices in other Cloud
  - On condition that other Cloud shall support OCF C2C API
  - Use OAuth 2.0 for Cloud Account Linking



C2C = Cloud to Cloud as defined by the OCF Cloud API for Cloud Services
D2C = Device to Cloud as defined by the OCF Device to Cloud Services Spec

Source: OCF_2.2.0_Specification_Overview.pptx

Core Framework : Ongoing Issues

- E2E Security based on OSCORE
- E2E Security – Multicast
- Onboarding with WiFi DPP
- Cloud Proxy
- Rules

IoTivity-Lite status

- Current Status

IoTivity

- Open Source project implementing OCF Core Framework
- IoTivity Classic : <= OCF 2.0.1
- IoTivity Lite : >= OCF 2.0.1
- https://iotivity.org/
- https://gitlab.iotivity.org/iotivity/iotivity-lite

IoTivity-Lite : Current Status

- The Latest release:
  - Support OCF 2.2.0 Core Framework
- Ipanema (next release of 2.2.0) release ready
  - E2E Security using OSCORE (RFC8613)
  - Secure Multicast UPDATE using OSCORE
- Microcontrollers
  - Support Arduino
  - Working on ESP32 (popular & inexpensive platform)
- C2C API Implementation (plgd Cloud Project)
  - https://github.com/plgd-dev/cloud
- DPP onboarding demo
  - OCF onboarding demo working (Cable Lab)
  - Presented in WFA (WiFi Alliance)